UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/039,777	12/31/2001	Deborah T. Marr	042390.P12495	9070

7590 05/21/2007

Jeffrey S. Draeger
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
Seventh Floor
12400 Wilshire Boulevard
Los Angeles, CA 90025-1026

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	**PAPER NUMBER**
2183	

MAIL DATE	DELIVERY MODE
05/21/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/039,777	MARR ET AL.	
	Examiner	Art Unit	
	David J. Huisman	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 March 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-6,8-14,16-20,22 and 23* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-6,8-14,16-20,22 and 23* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *8/10/06, 1/24/06, & 12/31/01* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-6, 8-14 and 16-20, and 22-23 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: Amendment and Extension of Time as received on 3/5/2007.

Claim Objections

3. Claim 8 is objected to because of the following informalities: Claim 8 is dependent on a canceled claim, and therefore, appropriate correction is required. For purposes of examination, the examiner will assume applicant meant claim 8 to be dependent on claim 1.

4. The "computer readable medium" of claim 12 currently encompasses non-statutory subject matter in the form of an optical or electrical wave. See page 14, paragraph [0031] of the specification. Therefore, applicant should amend the claim to say "The processor of claim 1 wherein said processor is embodied in digital format on a computer memory, a magnetic disc, or an optical disc." If applicant prefers alternative language comparable to that proposed by the examiner, said alternative language may be presented in an amendment for review by the examiner. If such an amendment is not made, the examiner will have to apply a 35 U.S.C 101 rejection in a subsequent Office Action.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-2, 8-10, 12, 18-19, and 22-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eickemeyer, U.S. Patent No. 6,931,639 (as applied in the previous Office Action), in view of Kalafatis et al., U.S. Patent No. 6,535,905 (as applied in the previous Office Action and herein referred to as Kalafatis).

7. Referring to claim 1, Eickemeyer has taught a processor comprising:

a) a plurality of thread partitionable resources that are each partitionable between a plurality of threads. See the abstract and column 1, lines 14-25, and note that multiple resources, such as a load queue, a store queue, etc., are partitioned among threads.

b) a plurality of shared resources shared by the plurality of threads including the first thread and at least one other thread. See Fig.2, and notice at least components 202, 238, 240, 242, and 244. For instance, all instructions (regardless of thread) would be stored in instruction cache 202. Similarly, execution units are used to execute all threads.

c) logic to receive a program instruction from a first thread directing said processor to suspend execution of said first thread, and in response to said program instruction to cause the processor to suspend execution of the first thread, to relinquish portions of said plurality of thread-partitionable resources associated with the first thread for use by at least one other thread of said

plurality of threads. See column 5, lines 5-10, and note that during operation, a thread may be removed (suspended) by executing an instruction which results in suspension. Also, see column 4, lines 12-30, and note that when a thread is suspended, the remaining thread is able to use the suspended thread's resources (i.e., they're relinquished by the suspended thread).

d) Eickemeyer has not taught that said plurality of thread partitionable resources comprise an instruction queue and a register pool. However, recall that Eickemeyer has taught the relinquishing of a partition of shared resources such as a load queue, store queue, and completion queue. See column 1, lines 14-25 and column 4, lines 12-30. Kalafatis has taught that multiple threads may share a register file and an instruction queue as well. See Fig.4, component 103, and column 13, lines 32-38. One would be motivated to make the register file and instruction queue of Kalafatis partitionable because Eickemeyer has disclosed that partitioned resources can result in higher performance and increased flexibility (as opposed to simply duplicating resources). See column 1, lines 28-37. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer such that the register file and instruction queue are partitioned (shared). Furthermore, the relinquishing of partitions in these resources would be obvious for the same reasons partitions in the load, store, and completion queues are relinquished (to give the processing thread more resources to work with).

8. Referring to claim 2, Eickemeyer in view of Kalafatis has taught a processor as described in claim 1. Eickemeyer has further taught that the program instruction is a suspend instruction which consists of a suspend opcode which explicitly directs the processor to suspend execution of the first thread and to relinquish portions of said plurality of thread partitionable resources associated with the first thread for use by other ones of said plurality of threads. Again, see

column 5, lines 5-10, and column 4, lines 12-30, and note that an instruction may be used for suspension. Instructions inherently have opcodes to tell the system which operation to perform.

9. Referring to claim 8, Eickemeyer in view of Kalafatis has taught a processor as described in claim 1. Eickemeyer has further taught that the processor further comprises:

a) the plurality of shared resources, said plurality of shared resources comprising:

a1) a plurality of execution units. See Fig.2, components 238-244.

a2) a cache. See Fig.2, component 246.

a3) a scheduler. A scheduler is inherent as something must decide which instructions are to be executed and when they are to execute.

b) a plurality of duplicated resources, said plurality of duplicated resources comprising:

b1) a plurality of processor state variables. See Fig.3, and note the duplicated thread registers which hold the state of a portion of the system.

b2) an instruction pointer. See Fig.2, component 206. Note that multiple instruction fetch address registers exist (one for each thread).

b3) register renaming logic. See Fig.2, component 234. If there are multiple threads executing at once, and renaming occurs for each, then there must be sufficient renaming resources for both threads.

10. Referring to claim 9, Eickemeyer in view of Kalafatis has taught a processor as described in claim 8. Eickemeyer has further taught that said plurality of thread partitionable resources further comprises: a plurality of re-order buffers and a plurality of store buffer entries. See column 1, lines 14-40, and column 4, lines 12-30, and note that a store queue and completion table (reorder buffer) have partitions relinquished.

11. Referring to claim 10, Eickemeyer in view of Kalafatis has taught a processor as described in claim 1. Eickemeyer has further taught that said logic is further to cause the processor to resume execution of said first thread in response to an event. See column 5, lines 5-7. Note that a thread may be added to a processor in response to an instruction (event). Or, a stalled thread could be resumed in response to an action being satisfied (for instance, a cache miss being resolved).

12. Referring to claim 12, Eickemeyer in view of Kalafatis has taught a processor as described in claim 1. Eickemeyer has not taught that said processor is embodied in digital format on a computer readable medium. However, Official Notice is taken that hardware, which is taught by Eickemeyer (Fig.2) and software are logically equivalent. That is, anything implemented in hardware may also be implemented in software and vice-versa. Designers with different goals make different decisions. Furthermore, when designing a processor, the processor design may be simulated in software before the hardware is built. This allows for quick modifications and savings in hardware costs. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer such that the processor is in digital format on a computer readable medium.

13. Referring to claim 18, Eickemeyer has taught a system comprising:

a) a memory to store a plurality of program threads, including a first thread and a second thread, said first thread including a first instruction. See Fig.2, component 202, and the abstract.

b) a processor coupled to said memory being separate from said processor, said processor including a plurality of thread partitionable resources and a plurality of shared resources, said processor to execute instructions from said memory, said processor, in response to execution of

said first instruction to suspend said first thread and to relinquish portions of said plurality of thread partitionable resources. See the abstract and column 1, lines 14-25, and note that multiple resources, such as a load queue, a store queue, etc., are partitioned among threads. Furthermore, see Fig.2, and notice at least components 202, 238, 240, 242, and 244. Finally, see column 5, lines 5-10, and note that during operation, a thread may be removed (suspended) by executing an instruction which results in suspension. Also, see column 4, lines 12-30, and note that when a thread is suspended, the remaining thread is able to use the suspended thread's resources (i.e., they're relinquished by the suspended thread).

c) Eickemeyer has not taught that said plurality of thread partitionable resources comprise an instruction queue and a register pool. However, recall that Eickemeyer has taught the relinquishing of a partition of shared resources such as a load queue, store queue, and completion queue. See column 1, lines 14-25 and column 4, lines 12-30. Kalafatis has taught that multiple threads may share a register file and an instruction queue as well. See Fig.4, component 103, and column 13, lines 32-38. One would be motivated to make the register file and instruction queue of Kalafatis partitionable because Eickemeyer has disclosed that partitioned resources can result in higher performance and increased flexibility (as opposed to simply duplicating resources). See column 1, lines 28-37. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer such that the register file and instruction queue are partitioned (shared). Furthermore, the relinquishing of partitions in these resources would be obvious for the same reasons partitions in the load, store, and completion queues are relinquished (to give the processing thread more resources to work with).

14. Referring to claim 19, Eickemeyer in view of Kalafatis has taught a system as described in claim 18. Eickemeyer has further taught that said processor is to execute said second thread from said memory while said first thread is suspended. See column 4, lines 12-30.

15. Referring to claim 22, Eickemeyer in view of Kalafatis has taught a system as described in claim 20. Eickemeyer has further taught that said processor further comprises:

a) a plurality of shared resources, said plurality of shared resources comprising:

a1) a plurality of execution units. See Fig.2, components 238-244.

a2) a cache. See Fig.2, component 246.

a3) a scheduler. A scheduler is inherent as something must decide which instructions are to be executed and when they are to execute.

b) a plurality of duplicated resources, said plurality of duplicated resources comprising:

b1) a plurality of processor state variables. See Fig.3, and note the duplicated thread registers which hold the state of a portion of the system.

b2) an instruction pointer. See Fig.2, component 206. Note that multiple instruction fetch address registers exist (one for each thread).

b3) register renaming logic. See Fig.2, component 234. If there are multiple threads executing at once, and renaming occurs for each, then there must be sufficient renaming resources for both threads.

16. Referring to claim 23, Eickemeyer in view of Kalafatis has taught a system as described in claim 22. Eickemeyer has further taught that said plurality of thread partitionable resources further comprises: a plurality of re-order buffers and a plurality of store buffer entries. See

column 1, lines 14-40, and column 4, lines 12-30, and note that a store queue and completion table (reorder buffer) have partitions relinquished.

17. Claims 3-6, 11, 13-14, 16-17, and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eickemeyer in view of Kalafatis and further in view of Emer et al., U.S. Patent No. 6,493,741 (herein referred to as Emer).

18. Referring to claim 3, Eickemeyer in view of Kalafatis has taught a processor as described in claim 1. Furthermore, while Eickemeyer has taught that said logic is to cause the processor to suspend the first thread (see column 5, lines 5-10), Eickemeyer has not explicitly taught that the first thread is suspended/removed for a selected amount of time. However, Emer has taught a type of suspend instruction (referred to as a quiesce instruction) which suspends a thread based on a timer value. See the abstract and Fig.8. Essentially, when a first thread in Emer is stalling, the quiesce instruction will cause suspension/removal of that thread so that it doesn't consume resources that non-stalling threads may be using. See column 2, lines 44-47, and column 4, lines 32-34. The timer is useful for preventing processor deadlock due to coding errors and harmful access violations. See column 9, lines 4-13. Also, one of ordinary skill in the art would have recognized that the timer ensures that a particular thread isn't starved. After a certain amount of time, each suspended thread would wake up. As a result, in order to at least prevent deadlocking in Eickemeyer, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer such that Eickemeyer's removal/suspend instruction is a quiesce instruction which causes a thread to be suspended for a selected amount of time, as taught by Emer.

19. Referring to claim 4, Eickemeyer in view of Emer has taught a processor as described in claim 3. Eickemeyer in view of Emer has further taught that said selected amount of time is a fixed amount of time. See column 4, lines 24-26, and claim 27 of Emer, and note that the timer is set to a predetermined time interval (fixed time).

20. Referring to claim 5, Eickemeyer in view of Emer has taught a processor as described in claim 3. Eickemeyer in view of Emer has further taught that said processor is to execute instructions from a second thread while said first thread is suspended. See column 4, lines 12-30, of Eickemeyer, and column 4, lines 32-34, of Emer.

21. Referring to claim 6, Eickemeyer in view of Emer has taught a processor as described in claim 3. Eickemeyer in view of Emer has not explicitly taught that said selected amount of time is programmable by at least one technique chosen from a set consisting of: providing an operand in conjunction with the program instruction, blowing fuses to set the selected amount, setting the selected amount in microcode. However, Official Notice is taken that processors may be implemented as PLDs and the like, where fuses are blown to achieve functionality. Consequently, when building a logic chip, the blowing of fuses is ultimately responsible for all delays and how the system operates. In this case, the timer value could be set based on fuses. Such devices allow a user to program the system in the desired manner, allowing for increased flexibility. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the processor of Eickemeyer in view of Emer as a PLD so that the selected amount of time is based on fuse blowing.

22. Referring to claim 11, Eickemeyer in view of Emer has taught a processor as described in claim 3. Eickemeyer in view of Emer has further taught that said logic is further to cause the

processor to ignore events until said selected amount of time has elapsed. When a thread is quiesced, as shown in Emer, all of its instructions (which cause events) are ignored until the timer expires. See Fig.8, steps 30, 32, and 36, and note that execution is resumed after the selected amount of time specified by the timer has elapsed.

23. Referring to claim 13, Eickemeyer has taught a method comprising:

a) receiving a first opcode in a first thread of execution. See column 5, lines 5-10, and note an instruction inherently having an opcode is received. Opcodes must exist to specify which operation is to be performed by the system.

b) suspending said first thread in response to said first opcode. See column 5, lines 5-10, and note that a thread may be removed/suspended.

c) relinquishing a plurality of thread-partitionable resources in response to said first opcode. See column 4, lines 12-30.

d) Eickemeyer has not taught suspending said first thread for a selected amount of time in response to said first opcode. However, Emer has taught a type of suspend instruction (referred to as a quiesce instruction) which suspends a thread based on a timer value. See the abstract and Fig.8. Essentially, when a first thread in Emer is stalling, the quiesce instruction will cause suspension/removal of that thread so that it doesn't consume resources that non-stalling threads may be using. See column 2, lines 44-47, and column 4, lines 32-34. The timer is useful for preventing processor deadlock due to coding errors and harmful access violations. See column 9, lines 4-13. Also, one of ordinary skill in the art would have recognized that the timer ensures that a particular thread isn't starved. After a certain amount of time, each suspended thread would wake up. As a result, in order to at least prevent deadlocking in Eickemeyer, it would

have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer such that Eickemeyer's removal/suspend instruction is a quiesce instruction which causes a thread to be suspended for a selected amount of time, as taught by Emer.

e) Eickemeyer in view of Emer has not taught that relinquishing said plurality of thread partitionable resources comprises relinquishing a partition of an instruction queue and relinquishing a plurality of registers from a register pool. However, recall that Eickemeyer has taught the relinquishing of a partition of shared resources such as a load queue, store queue, and completion queue. See column 1, lines 14-25 and column 4, lines 12-30. Kalafatis has taught that multiple threads may share a register file and an instruction queue as well. See Fig.4, component 103, and column 13, lines 32-38. One would be motivated to make the register file and instruction queue of Kalafatis partitionable because Eickemeyer has disclosed that partitioned resources can result in higher performance and increased flexibility (as opposed to simply duplicating resources). See column 1, lines 28-37. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer such that the register file and instruction queue are partitioned (shared). Furthermore, the relinquishing of partitions in these resources would be obvious for the same reasons partitions in the load, store, and completion queues are relinquished (to give the processing thread more resources to work with).

24. Referring to claim 14, Eickemeyer in view of Emer and further in view of Kalafatis has taught a method as described in claim 13. Eickemeyer has further taught that relinquishing comprises annealing the plurality of thread partitionable resources to become larger structures usable by fewer threads. See column 4, lines 12-30.

25. Referring to claim 16, Eickemeyer in view of Emer and further in view of Kalafatis has taught a method as described in claim 14. Eickemeyer has further taught that relinquishing said plurality of thread partitionable resources further comprises relinquishing a plurality of store buffer entries and relinquishing a plurality of reorder buffer entries. See column 1, lines 14-40, and column 4, lines 12-30, and note that a store queue and completion table (reorder buffer) have partitions relinquished.

26. Referring to claim 17, Eickemeyer in view of Emer and further in view of Kalafatis has taught a method as described in claim 13. Emer has further taught that said selected amount of time is programmable by at least one technique chosen from a set consisting of: providing an operand in conjunction with the first opcode; blowing fuses to set the selected amount of time; programming the selected amount of time in a storage location in advance of decoding a program instruction; setting the selected amount of time in microcode. Emer inherently teaches the third option as the timer will be set/programmed at some time T, which will be before decoding of some instruction at time T+X.

27. Referring to claim 20, Eickemeyer has taught a system as described in claim 19.

a) Eickemeyer has not taught that said processor is to suspend execution of said first thread in response to said first instruction for a selected amount of time. However, Emer has taught a type of suspend instruction (referred to as a quiesce instruction) which suspends a thread based on a timer value. See the abstract and Fig.8. Essentially, when a first thread in Emer is stalling, the quiesce instruction will cause suspension/removal of that thread so that it doesn't consume resources that non-stalling threads may be using. See column 2, lines 44-47, and column 4, lines 32-34. The timer is useful for preventing processor deadlock due to coding errors and harmful

access violations. See column 9, lines 4-13. Also, one of ordinary skill in the art would have recognized that the timer ensures that a particular thread isn't starved. After a certain amount of time, each suspended thread would wake up. As a result, in order to at least prevent deadlocking in Eickemeyer, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer such that Eickemeyer's removal/suspend instruction is a quiesce instruction which causes a thread to be suspended for a selected amount of time, as taught by Emer.

b) Eickemeyer in view of Emer has not taught that said selected amount of time is chosen by at least one technique chosen from a set consisting of: providing an operand in conjunction with the program instruction; blowing fuses to set the selected amount of time. However, Official Notice is taken that processors may be implemented as PLDs and the like, where fuses are blown to achieve functionality. Consequently, when building a logic chip, the blowing of fuses is ultimately responsible for all delays and how the system operates. In this case, the timer value could be set based on fuses. Such devices allow a user to program the system in the desired manner, allowing for increased flexibility. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the processor of Eickemeyer in view of Emer as a PLD so that the selected amount of time is based on fuse blowing.

Response to Arguments

28. Applicant's arguments filed on March 5, 2007, have been fully considered but they are not persuasive.

29. Applicant argues the novelty/rejection of claim 1 on pages 9-10 of the remarks, in substance that:

"Kalafatis is not describing relinquishing portions of thread partitionable resources..."

30. These arguments are not found persuasive for the following reasons:

a) The examiner agrees that Kalafatis does not describe relinquishing thread-partitionable resources. It is Eickemeyer that discusses relinquishing resources. However, Eickemeyer has only taught the relinquishing of some resources, which do not include a register pool and instruction queue. The question becomes, "Given both Eickemeyer and Kalafatis, would it have been obvious to one of ordinary skill in the art at the time of the invention to modify Eickemeyer in view of Kalafatis such that applicant's invention is realized?" The examiner asserts that the answer to this question is "yes." As pointed out in the rejection, Eickemeyer has taught the general concept that partitioning resources results in higher performance and increased flexibility. See column 1, lines 28-37. Kalafatis, meanwhile, has taught that a register pool and instruction queue are types of resources that may be shared among threads. The examiner believes that, given these two references and their teachings (specifically, Eickemeyer's teaching that partitioning shared resources is good, and Kalafatis' teaching that a register pool and instruction queue may be shared by threads), that it would have been obvious to modify the register pool and instruction queue of Eickemeyer to be shared, as taught by Kalafatis. Then, as shared resources, by partitioning them, flexibility and performance can be enhanced. Applicant should also realize that the reverse combination is also valid. That is, Kalafatis has taught partitioning of many resources among threads. Eickemeyer has taught the general idea of increasing flexibility and performance by relinquishing and/or repartitioning shared resources

among threads. Therefore, it would have also been obvious to one of ordinary skill in the art at the time of the invention to modify Kalafatis in view of Eickemeyer.

Conclusion

31. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Applicant is reminded that in amending in response to a rejection of claims, the patentable novelty must be clearly shown in view of the state of the art disclosed by the references cited and the objections made. Applicant must also show how the amendments avoid such references and objections. See 37 CFR § 1.111(c).

Chaudhry et al., U.S. Patent No. 7,168,076, has taught facilitating efficient join operations between a head thread and a speculative thread. Fig.16 shows a register file which is

partitioned into portions used by each thread, and when the speculative thread is complete, its registers are available to the head (primary) thread. See column 12, lines 1-29.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (571) 272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

DJH
David J. Huisman
May 14, 2007

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100